

## CORSO DI SISTEMI OPERATIVI A Prova del 7/10/2003

MATR. . . . . Cognome. . . . . Nome . . . . .

Username . . . . .

### NOTE

Il presente foglio va immediatamente compilato con le proprie generalità e matricola. E esso deve essere restituito al termine della prova. In caso di mancata restituzione, la prova dello studente non verrà presa in considerazione per la correzione.

### IMPORTANTE

Tutti i file sorgenti prodotti dallo studente per l'esame devono essere memorizzati in un direttorio denominato **soa-071003-x** nella propria home, dove **x** rappresenta il carattere **i** per gli Informatici, **t** per i Telecomunicazionisti, **e** per gli Elettronici. Soluzioni contenute in altri direttori non verranno prese in considerazione per la correzione.

## Prova UNIX-1

Si realizzi in ambiente Unix/C la seguente interazione tra processi:

- un processo deve essere invocato con un parametro sulla linea di comando (un numero intero) che indica quanti processi figli deve creare;
- dopo un'attesa di durata casuale tra 1 e 5 secondi, ogni processo figlio invia al padre un segnale SIGUSR1;
- il padre deve attendere i segnali inviati dai figli ;
- quando il padre ha ricevuto tutti i segnali dai figli, termina.

Devono essere utilizzate le primitive per la gestione affidabile dei segnali.

## Soluzione

```
#include <signal.h>
#include <stdlib.h>
#include <stdio.h>
```

```
void sigusr1_handler(int signo)
```

```
{  
  
}
```

```
main(int argc, char *argv[])  
{  
    int pid,i,n;  
    struct sigaction act;  
    sigset_t sigmask, zeromask;  
  
    /* CONTROLLO ARGOMENTI */  
  
    if(argc !=2)  
    {  
        fprintf(stderr,"Uso %s N\n",argv[0]);  
        exit(-1);  
    }  
  
    /* GESTIONE SEGNALI */  
  
    sigemptyset( &zeromask);  
  
    sigemptyset( &sigmask);  
    sigaddset(&sigmask, SIGUSR1);  
  
    /* Blocco del segnale SIGUSR1 */  
    sigprocmask(SIG_BLOCK, &sigmask, NULL);  
  
    act.sa_handler= sigusr1_handler;  
    sigemptyset( &act.sa_mask);  
    act.sa_flags= 0;  
  
    sigaction(SIGUSR1, &act, NULL);  
  
    /* CREAZIONE FIGLI */  
  
    n = atoi(argv[1]);  
  
    if(n<1 ||n >10) /* Controllo del valore del parametro */  
        n=5 ;  
  
    for(i=0;i < n ;i++)
```

```

{
    if((pid=fork())<0)
    {
        perror("Creazione processo figlio (fork) :");
        exit(-2);
    }
    else
    {
        if(pid==0)          /* Processo figlio */
        {
            srand(getpid());    /* Per inizializzare il generatore di numeri casuali */
            sleep(rand()%5+1);  /* Attesa di durata casuale tra 1 e 5 secondi */

            kill(getppid(),SIGUSR1);

            printf("FIGLIO %d: Inviato il segnale SIGUSR1 al padre\n",getpid());
            exit(0);
        }
    }
}

/* PADRE: Attesa dei segnali dai figli */

for(i=0;i < n ;i++)
    sigsuspend(&zeromask);

printf("PADRE: Ricevuti tutti i segnali SIGUSR1 dai figli \n",getpid());

for(i=0;i < n ;i++)
    wait(NULL);

exit(0);
}

```